Applicant(s): ARIEH DON, ALEXANDER VEPRINSKY AND NATAN VISHLITSKY E30-042CON (99-200CON)
Serial No.: 09/541,158
Filed: March 31, 2000

In the Specification

Insert the following paragraph at page 1, line 1:

Cross Reference to Related Application

This application is a continuation of United States Patent Application Serial No. 09/541,158 filed March 31, 2000 entitled Method for Enhancing Host Application Performance With a DASD Using Task Priorities.

The paragraph beginning at Page 5, line 10 has been amended as follows:

Still another object of this invention is to provide a disk array storage device with improved performance for handling applications in which the completion of a write transaction to a logical volume acts as a control with respect to other transactions and tasks.

The paragraph beginning at Page 5, line 14 has been amended as follows:

In accordance with this invention a disk array storage device processes transactions with multiple tasks of different categories performed in the multiple logical storage devices according to the position of [according] task requests in a task queue. One logical storage device is assigned priority. Upon receiving a task request from any of the logical storage

devices, this method determines the task category and logical storage device related to the task request. If the task request is for a task in a first category, it is assigned to the task queue in a position having a first priority. If the task request corresponds to a task in a second category, the task request transfers to the task queue in a position having a second, lesser priority. All other task requests are transferred to the task queue at positions of a third priority that is less than the second priority.

The paragraph beginning at Page 7, line 8 has been amended as follows:

In accordance with another aspect of this invention, the performance of an e-mail application program operating in a data processing system with a host processor and a disk array storage device is enhanced. The disk array storage device includes logical storage devices that operate in response to [task] <u>tasks</u> including reconnect tasks and second and third categories of other tasks. One logical storage device is dedicated to store a log. In accordance with this method, a task queue is maintained for different task requests that schedule the order by which the tasks are completed within the disk array storage device. Upon receiving a task request related to any of the logical storage devices, the method

initially determines the task category and the logical storage device related to the task request. A reconnect task request related to a logical storage device other than the dedicated logical storage device transfers to a position in the task queue of a first priority. A reconnect task request related to the dedicated logical storage device transfers to a position in the task queue having the highest priority. Requests for tasks of the second category related to the dedicated logical storage device transfer to positions in the task queue of a second priority that is less than the first priority. Requests for any other tasks from the logical storage devices transfer to positions of a priority that is less than the second priority.

The paragraph beginning at Page 8, line 14 has been amended as follows:

In accordance with another aspect of this invention the performance of an e-mail application operating in a data processing system of the host processor in a disk array storage device is improved. The disk array storage device comprises a plurality of logical storage devices that operate in response to tasks including reconnect tasks generated when a logical storage device completes a task, a second category of tasks generated in response to certain activities within the disk array storage device and a third category of other tasks. In

the e-mail system each e-mail the creation of each new e-mail message will require a log entry in a dedicated logical storage device as a condition precedent to the processing of any transaction. In accordance with the method a first- []in and first- []out task queue is maintained for different task requests and schedules the order by which the corresponding tasks are completed within the disk array storage device. The system operates by normally adding reconnect tasks at the beginning of the queue and other tasks to the end of the task queue. However, in an e-mail system the system transfers reconnect tasks related to the dedicated logical storage device to the head of the task queue whereby reconnect tasks related to other logical storage devices are transferred to a location following a reconnect task in the queue related to the dedicated logical storage device. Task requests of the second category related to the dedicated logical storage device transfer to a position in the task queue that follows any reconnect task requests in the task queue.

The paragraph beginning at Page 11, line 15 has been amended as follows:

A bus 19, typically with a SCSI interface, connects the server 12 to a disk array storage device 20. For purposes of explaining this invention, this disclosure describes a

Symmetrix disk array storage device sold by the assignee of this application. FIG. 1 depicts a general configuration of the disk array storage device as including a host adapter (HA) 21 and a cache memory 22 connected over a bus 23. A disk array also connects to the bus 24 and includes a number of disk adapters and physical disk drives. FIG. 1 depicts, in particular, a disk adapter (DA) 25 with a plurality of physical disk drives 26, a disk adapter (DA) 27 with a plurality of physical disk drives 28 and a disk adapter (DA) 30 with a plurality of physical disk drives 31. A system manager 32 connects through one of the disk adapters, such as the disk adapter 30, for controlling the configuration of the disk array storage device 20, all as known in the art. Each of the adapters, such as the host adapter 21 and disk adapters 25, 27 and 30, operates under the control of a microprocessor based system.

The paragraph beginning at Page 15, line 15 has been amended as follows:

Referring to FIG. 2, the host adapter 21 has access to a configuration buffer 41 with an entry for each logical storage device. The configuration buffer 41 may be located within the host adapter [20] 21 or within the cache memory 22. Each entry has a number of information items pertaining to a particular

logical device, such as a logical storage device identification (ID), a PRIORITY DEVICE flag, a WRITE SPACE ON flag and a WRITE SPACE MAX field. The PRIORITY DEVICE flag identifies the one logical storage device that is to be given priority over all other logical storage devices. In this particular embodiment, the LV-LOG logical storage device is the device that is given priority so its corresponding PRIORITY DEVICE flag is set. The method of setting a flag in a configuration buffer or other buffer is well known in the art. Setting the PRIORITY DEVICE flag in the configuration buffer 41 for the LV-LOG logical storage device constitutes assigning priority to one of the logical storage devices. The function of the WRITE SPACE ON flag is set forth more clearly later. However, typically the WRITE SPACE ON flag will be set only if the corresponding PRIORITY DEVICE flag is set.

The paragraph beginning at Page 16, line 12 has been amended as follows:

FIG. 2 further depicts a monitor module [41] 41A in the control 40. This monitor module [41] 41A takes the form of a program that operates in the control 40 to determine whether any task requests are pending for any logical storage devices or other devices in the configuration buffer 41. When a task request exists, the monitor module [41] 41A places it on a task

request queue 42 at a position identified by a task request pointer 43. FIG. 2 also depicts a task request generator 44 that represents the means by which various task requests are generated. Although shown as a discrete module, it will be apparent that such a module will be implemented in software. FIG. 2 also depicts a plurality of command queues 45. Each command queue receives commands from the host processor directed to a corresponding logical storage device. [. ] The function of a priority counter 46 is described later.

The paragraph beginning at Page 17, line 9 has been amended as follows:

FIGS. 3A and 3B depict the operation of the monitor module [41] 41A in response to the receipt of a task request at step 63. A series of steps 64, 65 and 66 provide a decoding function by determining the category of the requested task and the logical storage device with which it is related. In accordance with this invention, one of the logical devices is designated as a priority device. To enhance the operation of an e-mail system, the log file logical storage device, such as the LV-LOG logical storage device 26 in FIG. 1, is given priority.

The paragraph beginning at Page 17, line 18 has been amended as follows:

Reconnect task requests constitute a first category of task requests. In a disk array storage device as available from the assignee of this invention, reconnect tasks are always placed at the top of the task request queue 42 and positioned at a first or highest priority. In accordance with this invention, however, a distinction is made between those reconnect tasks from the priority device and those reconnect tasks from all other devices. If the reconnect task is related to a priority device, such as the LV-LOG logical storage device, steps 64 and 65 direct control to step 67 that sets an HOQ flag [as shown in FIG. 3B]. Control then transfers to step 70 in FIG. 3B that sets various pointers 43 to the task request queue 42. If the HOQ flag is set, as it is when the reconnect task request relates to a priority device, step 71 transfers control to step 72 that places this reconnect task request at the head of the task request queue [41]42, i. e., at the position of highest priority. Then the monitor module [41] 41A performs various housekeeping operations in step 73 to restore any registers. Thus, any time the host adapter 21 in FIG. 1 recognizes a reconnect task request from a priority device, it enables the corresponding reconnect task to be processed as a next task in sequence.

The paragraph beginning at Page 19, line 4 has been amended as follows:

If the reconnect task is from a non-priority device, control passes from step 65 in FIG. 3A to step 74 to begin a sequence that places a task in an appropriate position in the task request queue 42. As previously indicated, reconnect tasks generally have a high priority. Consequently a reconnect task from any non-priority device is still placed near the head of the task request queue. Specifically, step 74 determines if the first task request in the task queue is a reconnect task request. If it is not, then control passes back to the step 67 and this reconnect task request is placed at the head of the queue just as if it had come from the priority device.

The paragraph beginning at Page 19, line 15 has been amended as follows:

Step 74 then transfers control to a series of steps that analyze the task request queue 42 to determine an appropriate location for this priority task from a non-priority device dependent upon some independent operating condition, such as the number of task requests in the task request queue 42 in FIG. 2. For this specific independent operating condition step [. Step] 75 determines whether there are any other tasks in the queue. If the non-priority device reconnect task request being processed will be the only entry in the task request

queue 42, control passes directly to step 70 in FIG. 3B to place the task request on the task request queue 42. As it is the only task request, it will be the next task handled and, as it is a reconnect task request, if any other task requests arrive they will, with the exception of a reconnect task request from the priority device, be placed on the task request queue 42 after this task request.

The paragraph beginning at Page 20, line 10 has been amended as follows:

If the task request queue 42 contains task requests, step 75 transfers control to step 76 in FIG. 3A that determines the total number of task requests on the task request queue 42. Initially the number is greater than zero, so step 77 transfers control to step 80 that moves the task request pointers 43 to a next task on the task request queue 42. If this transfer causes the task request pointers 43 to go back to the start of the task request queue 42, step 81 transfers control to step 70 in FIG. 3B. Otherwise, step 82 determines whether the task in the location identified by the count is a reconnect task. If it is, step 83 decrements the task count and control transfers back to step 77. If a task request is received and all the tasks on the task request queue are reconnect tasks, control passes from step 77 to step 70 to load the task request at the

end of the queue. Thus, these steps transfer reconnect task requests related to the dedicated logical storage device to the head of the task queue whereby reconnect task requests related to other logical storage devices are transferred to a location following any reconnect task request related to the dedicated logical storage device.

The paragraph beginning at Page 22, line 3 has been amended as follows:

Looking first at a Start Command task request, steps 64 and 66 transfer control to step 85 that decodes the command and transfers control to step 86. Step 86 then monitors for the existence of a condition or completion of a control function as a condition precedent to any increase the priority of the Start Command task request from the priority device. In FIG. 3A the control function for this monitoring step 86 is represented as a "time to boost" test, and it uses the contents of the priority counter 46 shown in FIG. 2. The priority counter 46 controls the percentage of secondary category task requests from the priority device for which priority will be increased. That is, the priority counter 46 tracks the number of secondary category transfers from the dedicated logical storage device. The percentage is a function of the reciprocal of the number in the priority counter 46. That is, if every second category

command task request should be given priority, the priority counter 87 receives a "1". If 25% of the task requests are to receive priority, the priority counter receives "4". Whenever the value in the priority counter 46 is greater than 1, step 86 transfers to step 90 to decrement the counter and thereby load the Start Command task request at the bottom of the task request queue 42. When the priority counter 87 reaches a 1, however, control passes from step 86 to step 91 that reloads the priority counter with its initial number and thereafter enters the procedure beginning with step 74 to place the Start Command task request in the task request queue in locations of a second priority just following any reconnect tasks.

The paragraph beginning at Page 23, line 9 has been amended as follows:

As will now be apparent, if step 91 reloads the priority counter 46 with "1" the priority of all the second category tasks related to the priority device are enhanced. If the value is "2", 50% are enhanced. The number in the priority counter 46 will normally be "1" so that to enhance the priority of all second category requests from the priority device. However, the priority counter [42] <u>46</u> provides a means for preventing these tasks from excluding tasks from other logical storage devices to assure even system performance.

The paragraph beginning at Page 23, line 18 has been amended as follows:

A similar procedure monitors another condition as a condition precedent for increasing the priority of a WP Ceiling task request for the priority device. Steps 64, 66 and 85 transfer control to step 92 that [transfers] enables a priority change for write space task requests related to the dedicated logical storage device by shifting control to step 93 to determine whether the sequence should wait for write space. This depends upon the setting of a flag, such as the WRITE SPACE ON flag, shown in FIG. 2. As previously indicated, the WRITE SPACE ON flag will generally be set only for a priority device. In the flow diagram of FIG. 3A, the test of step 93 is not applied to non-priority devices. If the WRITE [SOURCE] SPACE ON flag is set, step 93 [transfers] is enabled to shift control to step 86. Otherwise the WP Ceiling task request is inserted at the end of the task request queue 42.

The paragraph beginning at Page 24, line 13 has been amended as follows:

_____Consequently, this process gives priority to second category task requests, such as the WP [ceiling] Ceiling and Start Command task requests, related to the priority device only if certain conditions are met. The "time to boost" test in

step 86, applies collectively to Start Command and WP Ceiling task requests and is one condition precedent to increasing task request priority. In the specific case of a WP Ceiling Task request, the WRITE SPACE ON flag must be set as a condition precedent for a priority increase for that task request.

The paragraph beginning at Page 25, line 4 has been amended as follows:

As will now be apparent, the priority device and other logical storage devices will only place one task request on the task request queue 42 at a time. However, certain task requests from the priority device will be taken up with enhanced priority by either being placed at, or transferred to, the top of the task request queue in the case of a reconnect task from the priority device or immediately following any reconnect tasks in the case of request of the second category such as the Start Command and WP Ceiling task requests. Otherwise all task requests from the priority device go to the bottom of the task request queue 42.

The paragraph beginning at Page 26, line 7 has been amended as follows:

This invention has now been disclosed in terms of a specific embodiment involving an e-mail system. It will be

apparent that many other variations could be incorporated. For example, the four-way decoder comprising steps 64, 65 and 66 decodes each task to determine whether it is a reconnect task and then determines whether the task is related to a priority device. This is shown by way of example for purposes of simplifying FIGS. 3A and 3B. The reverse approach of testing first to determine whether the task request is associated with a priority device and then determining the task is a reconnect task is actually used. Similarly, other sequences of tests could be incorporated while obtaining some or all of the advantages of this invention. In addition FIGS. 1 and 2 disclose a system with a single connection to a host and a single physical disk drive that stores a log file. Some disk array storage systems may include multiple paths or ports to different host systems [on] or different host applications. If multiple ports exist, the priority device will be assigned only for transactions from a corresponding port. Also in some systems, the physical disk drives may be configured so that the priority device appears as a single logical storage device to the host application, even though log entries are stored on multiple physical disk drives. The foregoing description of an e-mail system assumes that the log device will be the only priority device accessible to host applications. In some other applications it may be desirable to identify multiple logical

storage devices or other logical devices as priority devices for enhancing host application performance. Performance issues for a specific host application will dictate the identity of a priority device or priority devices and the identity of tasks for each category.